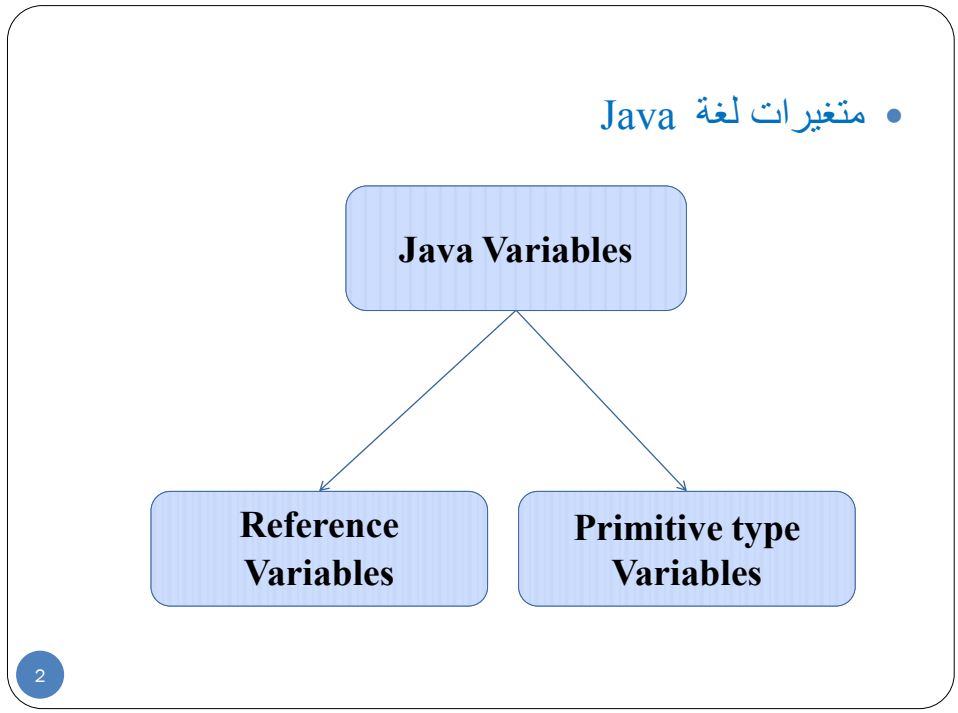
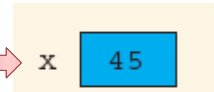

جامعة طرابلس
كلية تقنية المعلومات
خريف 2019
البرمجة الشيئية
Object Oriented Programming
(with Java)
ITGS211
المحاضرة الخامسة
1

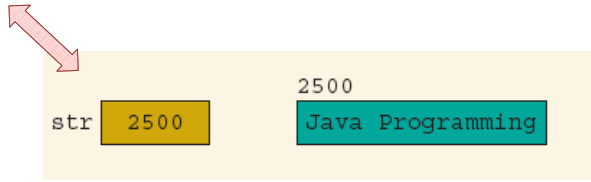

متغيرات لغة Java
Java Variables
Reference Variables
Primitive type Variables
2

Object and Reference Variables

المتغيرات من الانواع الأساسية Primitive تخزن البيانات مباشرة إلى موقعها في الذاكرة.
أما المتغيرات الـ Reference فهي تخزن عنوان الكائن الذي يحوي البيانات.

```
int x;      //Primitive variable
String str; //Reference variable
x = 45;     //x stores simple value
str = new String("Java Programming");
```



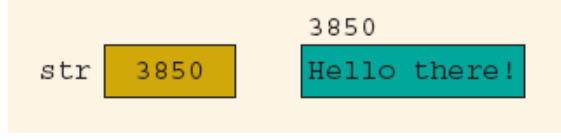


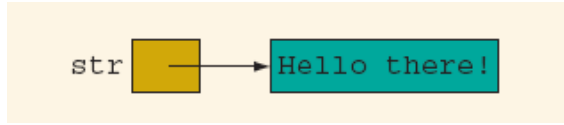
3

Object and Reference Variables

الكائن object ما هو الا صورة من الـ class والمؤثر new يستخدم لتكوين صورة من الكائن (instantiate an object)

```
str = new String("Hello there!");
```





The variable *references* the object

4

`String` Objects

- A variable can be assigned a string literal.

```
String value = "Hello";
```

`String` هو الـ object الوحيد الذي يُنشأ بهذه الطريقة

- A variable can be created using the *`new`* keyword.

```
String value = new String("Hello");
```

- هذه الطريقة التي يجب أن يُنشأ بها باقي الـ objects

- `String value = new String("Hello");`

تكافئ

- `String value = "Hello";`

```
String value;
value = "Hello";
```

5

`String` Methods

- The `String` class contains many methods that help with the manipulation of `String` objects.
- `String` objects are *immutable*, meaning that they cannot be changed.
- `String` class يحوي العديد من الـ methods التي تساعد في معالجة الكائن من نوع سلسة `String` objects والتعامل معه. فكائنات سلسلة هي ثابتة، وهذا يعني أنه لا يمكن تغييرها.

O	p	j	e	c	t		O	r	i	e	n	t	e	d
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

6

طرق معالجة الحرف Character Manipulation Methods

- يمكن معالجة حرف من حروف السلسلة باستخدام بعض الـMethods بدلا من كتابة برامج جديدة لتنفيذ هذه المعالجات، نذكر مها:

- `charAt(int index)`
  - مهمتها البحث عن الحرف الموجود في الموقع المُحدد بالدليل `index` من السلسلة، وارجاعه.
  - لاحظ أن:
    - قيمة الدليل لأول عنصر هي صفر 0.
    - قيمة الدليل `Index` يجب أن تكون محصورة بين صفر وطول السلسة

```
String s="Java";
char a;
a=s.charAt(0);
System.out.println("a=" + a);
```

Run → **a= J**

7

الاستبدال Replace

- `Srting replace(char ch1 ,char ch2)`
  - مهمتها استبدال حرف بحرف في سلسلة حرفيه حيث أن:
    - `ch1` الحرف المراد استبداله.
    - `ch2` الحرف المُراد الاستبدال به.
    - أي وضع الحرف `ch2` مكان `ch1`، وان تكرر يتكرر الاستبدال (أي كلما وجده استبدله)

```
s1=s.replace('v','*');
System.out.println(s1);
```

Run → **a= Ja*a**

```
s1=s.replace('a','*');
System.out.println(s1);
```

Run → **a= J*v***

8

```
public static void main(String[] args) {
    String s1="Plz. check ur Email";
    String s2="your";
    System.out.println(s1.length());
    System.out.println(s1.replace("ur",s2));
}
```

```
run:
19
Plz. check your Email
BUILD SUCCESSFUL (total time: 0 seconds)
```

9

أول ظهور indexOf & آخر ظهور lastIndexOf

- **indexOf** مهمتها البحث عن أول ظهور لقيمة المتغير Ch2 في السلسلة كما في المثال:

```
String s="How do you do";
int a;
a=s.indexOf('o');
System.out.println("a=" + a);
```

Run ⟵ **a= 1**

- **lastIndexOf:** مهمتها البحث عن آخر ظهور لقيمة المتغير Ch2 في السلسلة كما في المثال:

```
int a;
a=s. lastIndexOf('o');
System.out.println("a=" + a);
```

Run ⟵ **a= 12**

10

طول السلسلة length

مهمتها ارجاع قيمة عددية صحيحة تمثل طول السلسلة كما في المثال:

```
String s1,s="How do you do";
int a;
a=s.length();
System.out.println("a=" + a);
```

Run ⟹ a= 13

الوصل أو الربط concat

- مهمتها ضم أو ربط متغيرات من نوع سلسلة أو حرف مع بعضها البعض.

```
String I="Information", T=" Technology", F_name;
F_name =I.concat(T);
System.out.println("Faculty of:" + F_name);
```

```
run:
Faculty of:Information Technology
```

11

تبدأ بـ startsWith

تمكن من البحث عن الحرف أو الحروف التي تبدأ بها السلسلة، فإن طابق المطلوب بداية السلسلة، ترجع بـالقيمة `true` والا ترجع بالقيمة `false` كما في المثال:

```
String I="Information", T=" Technology";
System.out.println( I.startsWith("Inn"));
```

⟹ false

```
System.out.println( I.startsWith("In"));
```

⟹ true

تنتهي بـ endsWith

ترجع بـالقيمة `true` إذا طابق المطلوب نهاية السلسلة، والا ترجع بالقيمة `false` كما في المثال:

```
System.out.println( I.endsWith("tion"));
```

⟹ true

```
System.out.println( T.endsWith("tion"));
```

⟹ false

12

- **تغيير حالة حرف (أو أكثر) في سلسلة:**
- **إلى حروف كبيرة `toUpperCase ( )`:** تُمكن من تغيير حالة حرف أو أكثر في السلسلة من حروف صغيرة إلى حروف كبيرة.

```
String I="Information", T=" Technology";
System.out.println( I.toUpperCase()); ⟹ INFORMATION
```

- **إلى حروف صغيرة `toLowerCase ( )`:** تُمكن من تغيير حالة حرف أو أكثر في السلسلة من حروف كبيرة إلى حروف صغيرة.

```
System.out.println( T.toLowerCase()); ⟹ technology
```

13

- **مقارنة السلاسل:**
- **تساوي سلسلتين `equals ( )`:** ترجع بـالقيمة `true` في حالة تساوي السلسلتين تساوي تام حتى بالنسبة لحالة الحروف، والا ترجع بالقيمة `false` كما في المثال:

```
String I="Information", T="INFORMATION";
System.out.println( I.equals(T)); ⟹ False
```

- **تساوي سلسلتين مع تجاهل حالةالأحرف `equalsIgnoreCase ()`:** ترجع بـالقيمة `true` في حالة تساوي السلسلتين بغض النظر عن حالة الحروف سواء حروف كبيرة أو حروف صغيرة.

```
System.out.println( I.equalsIgnoreCase(T)); ⟹ true
```

14

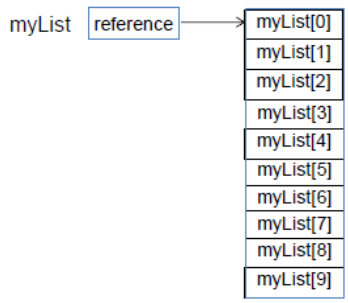
المصفوفات Arrays

- المصفوفات ذات البعد الواحد One-dimensional Arrays
- متعددة الأبعاد Multidimensional Arrays

المصفوفات ذات البعد الواحد
One-dimensional Arrays

المصفوفة: هي عبارة عن تركيبة من تراكيب البيانات لتمثيل مجموعة من البيانات ذات النوع الواحد.
بمعنى آخر هي مجموعة من خلايا الذاكرة المتجاورة تحت اسم واحد، وتحمل بيانات من نوع واحد. غالباً تكون عموداً واحداً

```
double[ ] myList = new double[10];  // مصفوفة أحادية بها 10 عناصر كسرية ذات دقة مضاعفة
```



وللتعامل مع أي عنصر من عناصر المصفوفة نستعمل اسم المصفوفة متبوعاً بالفهرس أو الدليل الخاص بذلك العنصر
مثلاً
myList[6]

الاعلان عن المصفوفة Array

- يكون الاعلان عن المصفوفة الأحادية بأحد الطريقتين:

- `datatype[] arrayname;`

Example:

```
double[] myList;
```

- أو

- `datatype arrayname[];`

Example:

```
double myList[];
```

مالفرق بينهما؟ ولماذا توجد طريقتين وليس طريقة واحدة؟

حيث أن :

`datatype`: نوع البيانات.
`arrayname`: اسم المصفوفة.

تكوين المصفوفة Creating

```
arrayName = new datatype[arraySize];
```

حيث :

`arrayname`: اسم المصفوفة.
`new`: كلمة محجوزة لتحديد أو تعيين مواضع allocation عناصر المصفوفة.
`datatype`: نوع البيانات.
`arraySize`: حجم المصفوفة (عدد عناصرها) ويكون عدد صحيح موجب.

مثال :

```
myList = new double[10];
```

`myList` مصفوفة أحادية مكونة من 10 عناصر من النوع الكسري double (أي من 0 إلى 9) بحيث أول عنصر يُشار إليه بـ `myList[0]` و آخر عنصر `myList[9]`

كما يمكن الاعلان عن المصفوفة وتحديد نوعها وتكوينها في خطوة واحدة كالتالي:

```
datatype[] arrayname = new datatype[arraySize];
Example: double[] myList = new double[10];
```

```
datatype arrayname[] = new datatype[arraySize];
Example: double myList[] = new double[10];
```

طول المصفوفة Length:

بالامكان تحديد طول المصفوفة باستخدام length.

myList.length

myList.length → 10

```
int m= myList.
```

length	int
hashCode()	int
clone()	Object
equals(Object o)	boolean
getClass()	Class<?>
notify()	void

```
int m= myList.length;
```

لا حظ:
length هي **خاصية** من خصائص المصفوفة.
بينما length() المستخدمة مع السلاسل الحرفية هي **دالة** ترجع طول السلسلة.

```
double myList []= new double [x];
int m= myList.length;
```

X

```
String y="Method";
int r =y.length();
```

تخصيص قيمة ابتدائية للمصفوفة

- باستخدام الحلقات كما يلي:

1)
```
for (int i = 0; i < myList.length; i++)
    myList[i] = i;
```

هنا كل عنصر قيمته تساوي قيمة دليل الحلقة (على سبيل المثال).

أو

2)
```
Scanner in= new Scanner (System.in);
for (int i = 0; i < myList.length; i++)
    myList[i] = in.nextDouble();
```

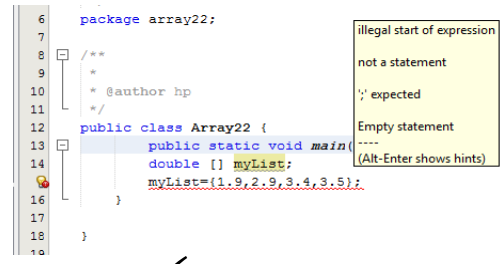
في حالة استقبال قيم المصفوفة من المستخدم

- بالطريقة المختزلة (shorthand notation) فيها يتم الاعلان عن مصفوفة وانشائها وتخصيص قيم ابتدائية لعناصرها في نفس الخطوة كما يلي:

```
double[ ] myList = {1.9, 2.9, 3.4, 3.5};
```

لاحظ : عند استخدام التدوين المختزل (Shorthand notation) عليك أن تعلن ،وتنشىء ،وتخصص قيم للعناصر في جملة واحدة كما في المثال السابق ولايمكن أن تجزء فعندها سينتج خطأ لغو ي:

```
double [ ] myList ;
myList={1.9,2.9,3.4,3.5};
```




```
double [ ] myList={1.9,2.9,3.4,3.5};
```

✓

```
*/
public class Array22 {
    public static void main(String[] args) {
    double [] myList ={1.9,2.9,3.4,3.5};
        System.out.println(myList[0]);
    }
```

22

```
public class Array22 {
        public static void main(String[] args) {
        double [] myList ={1.9,2.9,3.4,3.5};
            System.out.println(myList[0]);

    }
```

إن كتبنا السطر البرمجي التالي:
System.out.println(myList[4]);
هل ينتج خطأ؟ علل اجابتك مهما كانت
(سواءًا نعم أو لا)

23

برنامج يقوم بإدخال درجات عشر مواد لطالب وبعد ذلك يقوم البرنامج بكتابة مجموع الدرجات ومعدل الطالب؟

```
import java.util.Scanner;
public class Array1 {
   public static void main(String[] args) {
      Scanner in = new Scanner(System.in);
      int mark[]=new int[10];
      int sum =0; float average =0;
            for(int i=0; i<10;++i) {
               mark[i]=in.nextInt();
              sum=sum+mark[i];    }
      average=sum/10;
      System.out.println("sum is :"+sum);
      System.out.println("average is :"+ average +"%");
   }
```

```
run:
2
4
5
6
4
7
8
8
9
10
sum is :63
average is :6.0%
```

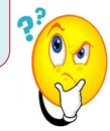
24

برنامج يقوم بعمل مصفوفة حروف تقوم بطباعة أيام الأسبوع ؟

```
public class Array1 {
   public static void main(String[] args) {
   String day[]={"Sat","Sun","Mon","Tus","wed","Thr","Fri"};
   for(int i=0; i<day.length;++i)
                System.out.println(day[i]);
}
```

```
run:
Sat
Sun
Mon
Tus
wed
Thr
Fri
```

ما ناتج نفس البرنامج في حال استبدال جملة الطباعة بالجملة:
System.out.println(day[i].length());



25

```
import java.util.Scanner;
public class TestArray {
    public static void main(String[] args){
      Scanner in = new Scanner(System.in);
      final int TOTAL_Size = 4;
      int[] numbers = new int[TOTAL_Size];
      for (int i = 0; i < numbers.length; i++){
          numbers[i] = in.nextInt();
      }
      int count=0, max = numbers[0];
      for (int i = 1; i < numbers.length; i++){
          if (max < numbers[i]){
              max = numbers[i];
              count=i;
          }
      }
      System.out.println(count);
      System.out.println(max);
    }
}
```

يقوم البرنامج باستقبال 4 أعداد صحيحة وايجاد أكبرها وتحديده موقعه في المصفوفة.

26

إذا كانت المصفوفة numbers كالتالي:

```
int numbers[ ] ={10,20,50,30,45}
```

ما ناتج جملة الطباعة التالية إذا كانت i تساوي 2:

```
System.out.println(numbers[i]+1);
```

الجواب هو يطبع **51** ولكن لماذا؟!!

```
System.out.println(numbers[2]+1);
```

→ 50 +1 =51

أمـــــا

```
System.out.println(numbers[i+1]);
```

فتطبع **30** أيضاً نسأل لماذا ☹؟!!

```
System.out.println(numbers[2+1];
System.out.println(numbers[3] =30
```

H. W.

أعد كتابة البرنامج في المثال السابق ليستقبل درجات 6 طلبة، ثم يقوم بطباعة المصفوفة بعد ترتيب الدرجات ترتيباً تصاعدياً، وطباعة أعلى درجة، طباعة متوسط هذه الدرجات.

اكتب برنامج يقوم بقراءة المصفوفة A والمصفوفة B ثم جمع العناصر المتناظرة في A وB وتخزينها في المصفوفة C، ثم طباعة المصفوفة C.

28

اكتب برنامج يقوم باستقبال n من القيم الصحيحة وتخزينها في المصفوفة A ثم طباعة مجموع القيم الموجبة S_Positive ومجموع القيم السالبة S_Negative

برنامج لقراءة عدد من السلاسل الحرفية وتخزينها في مصفوفة احادية ، ثم طباعتها بحروف كبيرة Capital letter .

```
/*** @author Nahed ***/
import java.util.Scanner;
public class ArrayOfString {
public static void main(String[] args) {
      Scanner input =new Scanner(System.in);
      System.out.println("أدخل حجم المصفوفة:");
       final int size=input.nextInt();                    //قراءة حجم المصفوفة
       String A[]= new String[size];                      //تكوين المصفوفة
      System.out.println("أدخل عناصر المصفوفة:");
       for(int i=0; i<A.length;i++){
          A[i]=input.next();
       }
       for(int i=0; i<size;i++){
          System.out.println( "A["+i+"]=\t"+A[i].toUpperCase());
       }
} }
```

قراءة عناصر المصفوفة

طباعة عناصر المصفوفة بعد تغيير حالة الأحرف إلى **capital letter**

30

هل فعلا تغيرت حالة أحرف عناصر المصفوفة واصبحت مكتوبة بحروف كبيرة؟

أعد البرنامج السابق مع طباعة طول كل عنصر من عناصر المصفوفة.

أعد البرنامج السابق مع البحث عن السلسلة "ITGS211" مع تجاهل مطابقة حالة الأحرف وطباعة موقعها.

31

برنامج لقراءة أسماء مجموعة من الطلبة وتخزينهم في مصفوفة أحادية، ومن ثم طباعة الحرف الأول من كل اسم.

```
import java.util.Scanner;
/***@author Nahed ***/
public class Array_charAt {
    public static void main(String[] args) {
        Scanner input =new Scanner(System.in);
        System.out.println("أدخل حجم المصفوفة:");
        final int size=input.nextInt();
        String A[]= new String[size];
        System.out.println("أدخل عناصر المصفوفة:");
        for(int i=0; i<A.length;i++){
            A[i]=input.next();
        }
        System.out.println("******************");
        for(int i=0; i<size;i++)
            System.out.println( "A["+i+"] starts with: "+A[i].charAt(0));
    }
}
```

```
run:
أدخل حجم المصفوفة:
3
أدخل عناصر المصفوفة:
klj
tre
wqa
******************
A[0] starts with: k
A[1] starts with: t
A[2] starts with: w
```

32

اكتب برنامج بلغة Java يستقبل درجات 6 طلبة، ويقوم بتخزينهم في مصفوفة Std أحادية، ومن تم طباعة:
- ✓أعلى درجة.
- ✓متوسط الدرجات.

```
package arrays;
import java.util.Scanner;
public class Arrays {
    public static void main(String[] args) {
      Scanner in = new Scanner(System.in);
      final int TOTAL_Size = 6;
      double[] Std= new double[TOTAL_Size];
      double sum=0, avrg;
      for (int i = 0; i < Std.length; i++){
          Std[i] = in.nextDouble();
          sum+=Std[i];
      }
      avrg=sum/TOTAL_Size;
      System.out.println("avrg=" + avrg);
      double max = Std[0];
      for (int i = 1; i < 6; i++){
          if (max < Std[i]){
              max =Std[i];
          }
        }
      System.out.println("Max=" + max);
        }
```

```
Output - Arrays (run)
run:
15
2
5
3
7
8
avrg=6.666666666666667
Max=15.0
```

33

اكتب برنامج بلغة Java يقوم بقراءة المصفوفة A والمصفوفة B، ثم يجمع العناصر المتناظرة في Aو B وتخزينهم في المصفوفة C، ثم طباعة المصفوفة

```
import java.util.Scanner;
public class Nahed_E1 {
    public static void main(String[] args) {
      Scanner in = new Scanner(System.in);
      final int TOTAL_Size = 4;
      int A[]= new int[TOTAL_Size];
      int B[]= new int[TOTAL_Size];
      int C[]= new int[TOTAL_Size];
      for (int i = 0; i < TOTAL_Size; i++)
      {
         System.out.print("Enter A["+ i + "]:");
         A[i] = in.nextInt();
          System.out.print("Enter B[" + i + "]:");
          B[i] = in.nextInt();
          C[i]=A[i]+B[i];
      }
      for (int i = 0; i < TOTAL_Size; i++)
     System.out.println("C["+i+"]="+ C[i]);

}
```

```
Output - Examples111 (run)
run:
Enter A[0]:1
Enter B[0]:1
Enter A[1]:5
Enter B[1]:5
Enter A[2]:12
Enter B[2]:12
Enter A[3]:4
Enter B[3]:4
C[0]=2
C[1]=10
C[2]=24
C[3]=8
BUILD SUCCESSFUL (total time: 14 seconds)
```

34

اكتب برنامج يقوم باستقبال n من القيم الصحيحة وتخزينها في المصفوفة A ثم طباعة مجموع القيم الموجبة S_Positive ومجموع القيم السالبة S_Negative

```
public class Nahed_E1 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int s_Positive= 0,s_Negative = 0;
        int n=in.nextInt();
        int A[]= new int[n];
        for (int i = 0; i < n; i++)
        {
          System.out.print("Enter A["+ i + "]:");
          A[i] = in.nextInt();
          if (A[i]>=0)
              s_Positive+=A[i];
          else
              s_Negative+=A[i];
        }
        System.out.println("s_Positive=" + s_Positive);
        System.out.println("s_Negative="+ s_Negative);
```

```
run:
5
Enter A[0]:2
Enter A[1]:-1
Enter A[2]:0
Enter A[3]:5
Enter A[4]:-2
s_Positive=7
s_Negative=-3
```

35

Foreach Loop

توفر أحدث إصدار Java نوعًا خاصًا من الحلقات لمعالجة المصفوفات، يُسمى foreach، فيها يتم تعريف متغير من نفس نوع المصفوفة وتأخذ الشكل العام التالي:

نوع بيانات المتغير وهو من نفس نوع بيانات عناصر المصفوفة

اسم المصفوفة

```
for (dataType identifier : arrayName)
    statements   // الجمل المراد تكرارها
```

حيث أن identifier: هو متغير من نفس نوع بيانات عناصر المصفوفة.

```
int B[]={10,30,50,70};
for(int i=0;i<B.length;i++){
    System.out.println(B[i]);
}
```

=

```
int B[]={10,30,50,70};
for(int N:B){
    System.out.println(N);
}
```

36

```
public class Array_for{
    public static void main(String[] args) {
    String day[]={"Sat","Sun","Mon","Tus","wed","Thr","Fri"}
    for(int i=0; i<day.length;++i)
        System.out.println(day[i]);
    }
}
```

متكافئتان ولهما نفس النتيجة

```
run:
Sat
Sun
Mon
Tus
wed
Thr
Fri
```

لاستخدام حلقة foreach يجب تعريف متغير من نفس نوع بيانات عناصر المصفوفة:
المصفوفة day أعلاه ↑ عناصرها من النوع String من هنا وجب تعريف متغير وليكن str من النوع String ، ويصبح المثال كالتالي:

```
public class Array_for{
    public static void main(String[] args) {
    String day[]={"Sat","Sun","Mon","Tus","wed","Thr","Fri"};
    for(String str:day)
        System.out.println(str);
    }
}
```

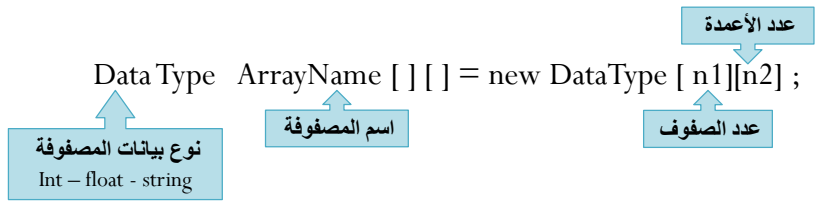
37

مصفوفات متعددة الأبعاد
Multidimensional Arrays

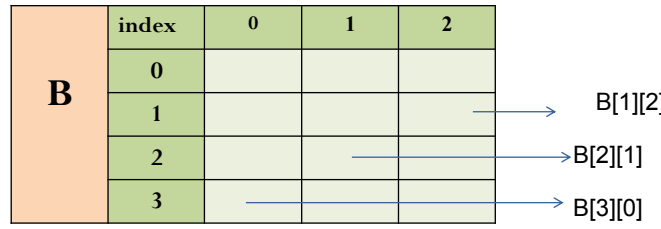
38

مصفوفات ثنائية الأبعاد Two Dimensional Array

المصفوفة ذات البعدين هي عبارة عن جدول يحتوي على صفوف وأعمدة
والصيغة العامة لهذه المصفوفة كالآتي :



Ex --->> Int B [ ] [ ] = new int [4][3] ;



39

الاعلان عن المصفوفة ذات البعدين وتكوينها

- يكون الاعلان عن المصفوفة (شبيه بالمصفوفة الأحادية) بأحد الطريقتين:

- `datatype[][] arrayname;`

Example:

```
double[][] myList;
```

- أو

- `datatype arrayname[][];`

Example:

```
double myList[][];
```

وانشاءها Creating (أيضا شبيه بالمصفوفة الأحادية) ويأخذ الشكل:

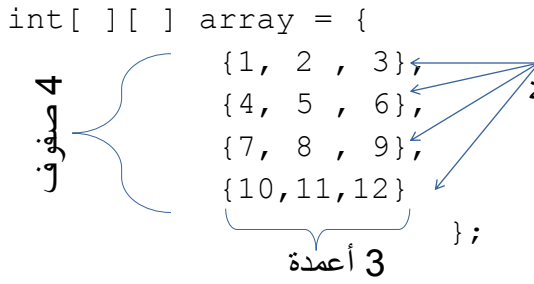
- `arrayName = new datatype[][];`

حيث أن :

`datatype`: نوع البيانات.
`arrayname`: اسم المصفوفة.

40

مصفوفات ثنائية الأبعاد Two Dimensional Array



في هذه المصفوفة لدينا 4 صفوف كل صف عبارة عن مصفوفة أحادية عدد عناصرها هو 3. أي مصفوفة 4*3 ← `array[4][3]`

41

مثال:

```
int arr2[][];
arr2 = new int [][]{{1,2,3},{4,5,6}};
```

كما يمكن الاعلان عن المصفوفة وتحديد نوعها وتكوينها في خطوة واحدة كالتالي:

- `int arr2[][]={{1,2,3},{4,5,6}};`

-
```
int[][] array = {  {1, 2, 3},
                   {4, 5, 6},
                   {7, 8, 9},
                   {10, 11, 12}
                };
```

في المصفوفة متعددة الأبعاد كل عنصر هو عبارة عن مصفوفة احادية.

-
```
int[][] array1 = {  {1, 2, 3},
                    {4, 5, 6, 7, 8},
                    {10}
                };
```

بما أن المصفوفة متعددة الأبعاد فيها كل عنصر هو عبارة عن مصفوفة احادية البُعد، قد تكون هذه المصفوفات الداخلية ذات اطوال مختلفة كما في هذا المثال.

42

طول المصفوفة ذات البُعدين Length

بالإمكان تحديد طول المصفوفة باستخدام الخاصية length التي سبق استخدامها مع المصفوفات أحادية البُعد

```
public static void main(String[] args) {
    int[ ][ ] array = {
                      {1, 2, 3},
                      {4, 5, 6},
                      {7, 8, 9},
                      {10, 11, 12}
                      };
  System.out.println(array.length);    → ?
  System.out.println(array[1].length); → ?
  }
```

43

طول المصفوفة ذات البُعدين

```
public static void main(String[] args) {
  int[ ][ ] array = {
array[0] ⟹    {1, 2, 3},
array[1] ⟹    {4, 5, 6},
array[2] ⟹    {7, 8, 9},
array[3] ⟹    {10, 11, 12}
                    };
  System.out.println(array.length);
```

عدد الصفوف
4 صفوف

مع اسم المصفوفة دون أقواس تعيد عدد صفوف المصفوفة، وهو 4 في هذا المثال

```
  System.out.println(array[1].length);
}
```

مع اسم المصفوفة وبين القوسين رقم الصف فإنها تعيد طول هذا الصف (أي عدد عناصر المصفوفة الأحادية)، وهو 3 في هذا المثال

44

طول المصفوفة ذات البُعدين

```
public static void main(String[] args) {
 int[ ][ ] matrix = {
        {1, 2, 3, 4, 5},
        {2, 3, 4, 5},
        {3, 4, 5},
        {4, 5},
        {5}
      };
   System.out.println(matrix.length);

   System.out.println(matrix[1].length);

   System.out.println(matrix[4].length);
```

5 صفوف
الصف 0
الصف 1
الصف 2
الصف 3
الصف 4

في المصفوفة ذات البعدين كل عنصر هو عبارة عن مصفوفة احادية البُعد، قد تكون هذه المصفوفات ذات اطوال مختلفة كما في هذا المثال.

Length مع اسم المصفوفة بدون أقواس تعيد عدد صفوف المصفوفة، وهو هنا 5

هنا يُعيد طول المصفوفة matrix[1] التي في الصف الثاني وهو 4

هنا يُعيد طول المصفوفة matrix[4] التي في الصف الخامس وهو 1 عنصر واحد

45

قراءة عناصر المصفوفة ذات بعدين

```
public static void main(String[] args) {
  Scanner in= new Scanner (System.in);
  int arr2[][];
     arr2 = new int [3][3];

     for(int i=0; i<3;i++)
     {
        for(int j=0; j<3;j++)
           arr2[i][j]=in.nextInt();
     }
}
```

الاعلان عن المصفوفة ذات البعدين وتكوينها

يتم استقبال البيانات من المستخدم وتخزينها في المصفوفة باستخدام حلقتين متداخلتين.
(الأولى للصفوف والثانية للأعمدة)

46

المثال السابق واستخدام الخاصية length

```
public static void main(String[] args) {
 Scanner in= new Scanner (System.in);
 int arr2[][];
       arr2 = new int [3][3];

    for(int i=0; i< arr2.length ;i++)
     {
        for(int j=0; j<arr2[i].length;j++)
          arr2[i][j]=in.nextInt();
     }
}
```

الاعلان عن المصفوفة ذات البعدين وتكوينها

Length مع اسم المصفوفة بدون أقواس تعيد عدد الصفوف، وهو هنا 3

هنا كل مرة يُعيد طول المصفوفة arr2[i]

تتم قراءة المصفوفة (استقبال البيانات من المستخدم) وتخزينها في المصفوفة ذات بعدين باستخدام حلقتين متداخلتين. (الأولى للصفوف والثانية للأعمدة)

47

طباعة عناصر المصفوفة ذات بعدين

```
public static void main(String[] args) {
  Scanner in= new Scanner (System.in);
  int arr2[][];
     arr2 = new int [3][3];

  for(int i=0; i<3;i++)
   {
     for(int j=0; j<3;j++)
       arr2[i][j]=in.nextInt();
   }

   for(int i=0; i<3;i++)
  {
     for(int j=0; j<3;j++)
        System.out.print(arr2[i][j]+ " ");
      System.out.println();
   }
}
```

القراءة باستخدام حلقتي for

الطباعة باستخدام حلقتي for

```
run:
4
4
6
8
9
7
6
4
3
4  4  6
8  9  7
6  4  3
```

48

طباعة عناصر القطر الرئيسي لمصفوفة ذات بعدين

```
public static void main(String[] args) {
   int[ ][ ] array3 = { {1, 2, 3 , 20},
                        {4, 5, 6 , 21},
                        {7, 8, 9, 22 },
                        {10, 11, 12, 23}
                      };
```

```
run:
1
5
9
23
```

الطباعة باستخدام حلقتي for

```
   for(int i=0; i<4;i++)
   {
     for(int j=0; j<4;j++)
        if (i==j) {
            System.out.println (array3[i][j]+ " ");
          }
   }
}
```

أعد كتابة البرنامج السابق ليقوم بطباعة عناصر القطر الثانوي للمصفوفة ؟

49

طباعة الجدر التربيعي لعناصر القطر الرئيسي

```
public static void main(String[] args) {
 int[][] array3 = {
                   { 1,  3  ,  5 ,  7 },
                   { 2,  4  ,  6  , 8 },
                   {15, 20 , 25, 30},
                   { 3,  5  ,  7,   9 }
                  };
     for(int i=0; i<4;i++)
     {
       for(int j=0; j<4;j++)
          if (i==j){
             System.out.print(Math.sqrt(array3[i][j]));
             System.out.println();
             }
     }
  }
```

باستخدام الدالة الرياضية الجاهزة Sqrt

```
Math.sqrt(array3[i][j])
```

فيكون ناتج تنفيذ البرنامج

```
1.0
2.0
5.0
3.0
```

50

طباعة عناصر المثلث السفلي للمصفوفة (العناصر التي أسفل القطر الرئيسي)

```
public static void main(String[] args) {
 int[][] array3 = {
                   { 1,  3  ,  5 ,  7 },
                   { 2,  4  ,  6  , 8 },
                   {15, 20 , 25, 30},
                   { 3,  5  ,  7,   9 }
                  };
        for(int i=0; i<4;i++)
       {
          for(int j=0; j<4;j++)
             if (i>j)
                System.out.print(array3[i][j] + "  ");
          System.out.println();
       }
```

ناتج تنفيذ البرنامج:

```
 2
15  20
 3   5    7
```

أعد كتابة البرنامج السابق ليقوم بطباعة عناصر المثلث العلوي للمصفوفة ؟

51

برنامج يعمل على ادخال درجات 6 مواد لأربع طلاب ثم يطبع الدرجات والمعدل لكل طالب ؟

```
package array5;
import  java.util.Scanner;
public class Array5 {
    public static void main(String[ ] args) {
        Scanner input= new Scanner(System.in);
        int student[ ][ ]=new int[4][7];
        int i,j;
        for(i=0;i<4;++i){
            int sum=0 ;
            for(j=0;j<6;++j){
                student[i][j]=input.nextInt();
                sum+=student[i][j];
                if(j==5)
                    student[i][6]=sum/6;}
                    }
```

52

```
for(i=0;i<4;++i){
        System.out.print("student No("+i+")");
        for(j=0;j<6;++j){
            System.out.print("\t"+student[i][j]);
            if(j==5)
            System.out.println("\t Average is"+student[i][6]);
                    }
     }
  }
```

```
student No(0)   68      88      93      98      74      83      Average is84
student No(1)   92      95      94      99      73      77      Average is88
student No(2)   83      90      92      75      88      95      Average is87
student No(3)   93      96      92      84      85      74      Average is87
```

53

انتهت المحاضرة

54